Requirements Spec Revisited

Dan Fleck

Functional Requirements

- A functional requirement is something the system must do.
- A functional requirement is testable
- A general rule is a functional requirement is a “shall statement”
  - The system shall require users to login to access all functions.
- A single use case will generate many functional requirements. Think of all the little steps needed for your system to satisfy the use case. Include requirements for alternate and exception flows!

Functional Requirements

- These can be high level or low level (generally we’re at high level in this class)
  - High level: The system shall charge users credit cards for purchases
  - Low level: The system shall validate all passwords contain upper and lowercase characters and one number

Functional Requirements

- Are testable
- Are things the system you are developing must do
- Should be one thing (not multiple). (Because a requirement is a single entity… it passes or fails as one piece)
- Should have a source (who/what decided this was required)

Example

- Requirements must do ONE THING.
  - Bad: The system shall accept credit cards and accept pay pal
  - Good:
    - The system shall accept credit cards
    - The system shall accept pay pal.

- Requirements must be testable. Use precise language.
  - Bad: The system shall work with any browser
  - Good:
    - The system shall work with Firefox
    - The system shall work with IE
  - Bad: The system shall respond quickly to user clicks
  - Good: The system shall respond within 10ms to any user click

Functional Requirements

- Should not be a design choice (this is hard to get right).
  - The system shall store user information including name, DOB, address and SSN. <-- Good!
  - The system shall store user information in an Oracle database including name, DOB, address, SSN. <-- **bad**
    - Is Oracle really REQUIRED? Hard to say… maybe, but probably not. This is a decision you would make at implementation design time.
    - Question: Does the customer care that you use Oracle? MySQL? Etc.. Maybe someone found some other MUCH BETTER approach storing the data on moon rocks.
    - Again: This is hard to avoid… and I’m not to concerned with it on the SRS, but I want you to be very aware of when you are making design choices instead of required features.

Functional Requirements

- Must have a unique ID.
  - When testing you need to reference REQ-1 or REQ-287. Multiple things cannot be labeled REQ-1.
  - Later our test cases will say: This test case validates requirements REQ-1, REQ-27, and REQ-56.

Functional Requirements

- **Bad requirements examples:**
  - The system shall validate and accept credit cards and cashier’s checks. High priority.
  - The system shall process all mouse clicks very fast to ensure user’s do not have to wait.
  - The user must have Adobe Acrobat installed.

Functional Requirements

- **Bad requirements examples:**
  - The system shall validate and accept credit cards and cashier's checks. High priority.
    - Problem: two requirements instead of one.
    - If the credit card processing works, but the cashier's check validation does not… is this requirement pass or fail? Has to be fail, but that is misleading.
    - Maybe only credit cards are high priority and cashier's checks are low priority.
  - The system shall process all mouse clicks very fast to ensure user's do not have to wait.
    - Problem: This is not testable. Quantify how fast is acceptable?
  - The user must have Adobe Acrobat installed.
    - Problem: This is not something our system must do. It could be in the constraints/assumptions or maybe operating environment sections, but is not a functional requirement of our system

Requirements Chart

ID	Priority	Type F = Functional NF = Non-Functional	Source	Contained in Use Case(s)	Description
3	High	F	Customer - John Smith	U3, U8	The system shall provide an option to withdraw money
3.1	Medium	F	Customer - John Smith	U3, U8, U10	The system shall query the user for the amount of money
1	High	F	Internal Team	U1	The system shall require user login before any operation
1.1	Medium	F	Internal Team	U1	The system shall lock users out who have failed the maximum number of password attempts